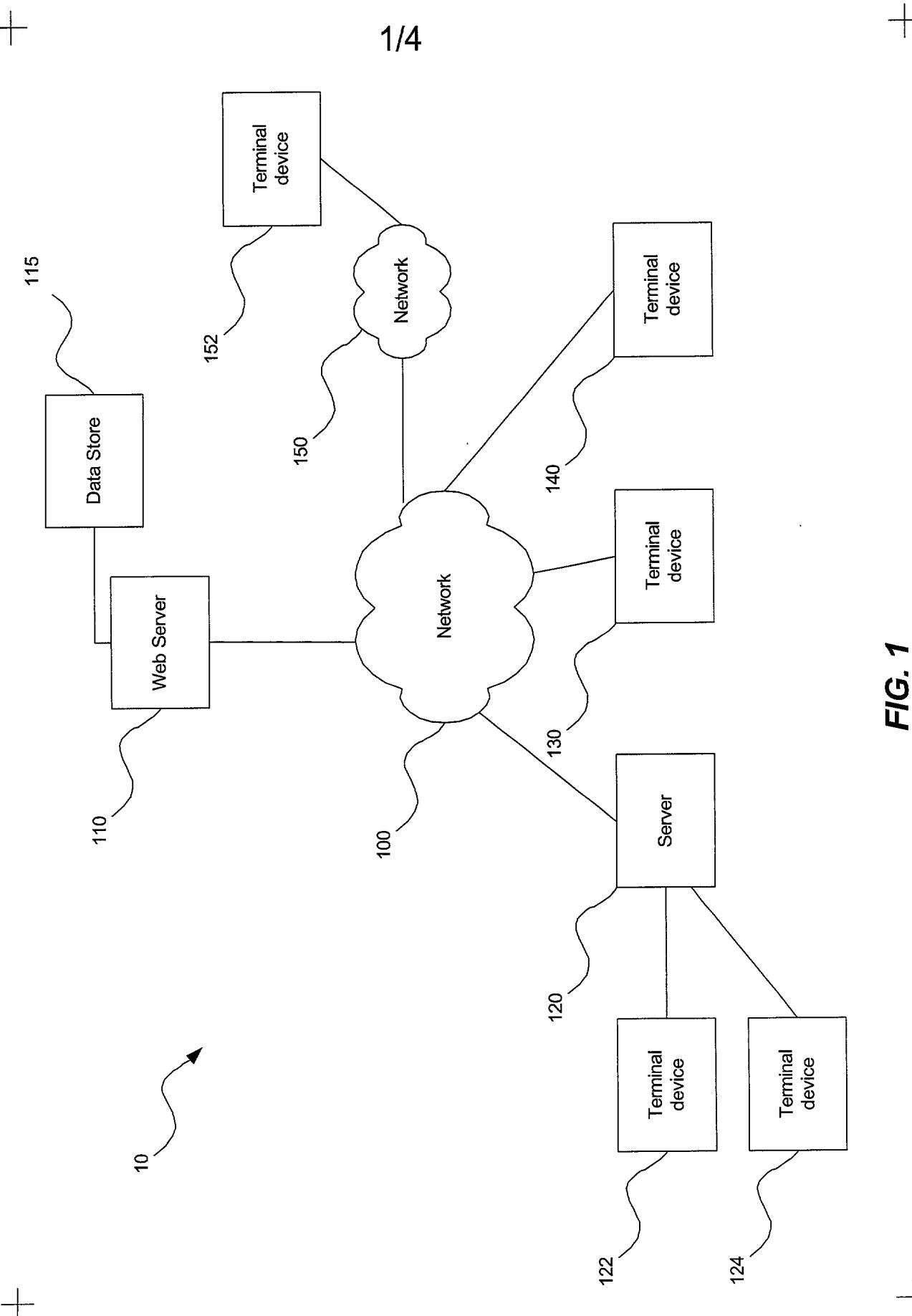


FIG. 1



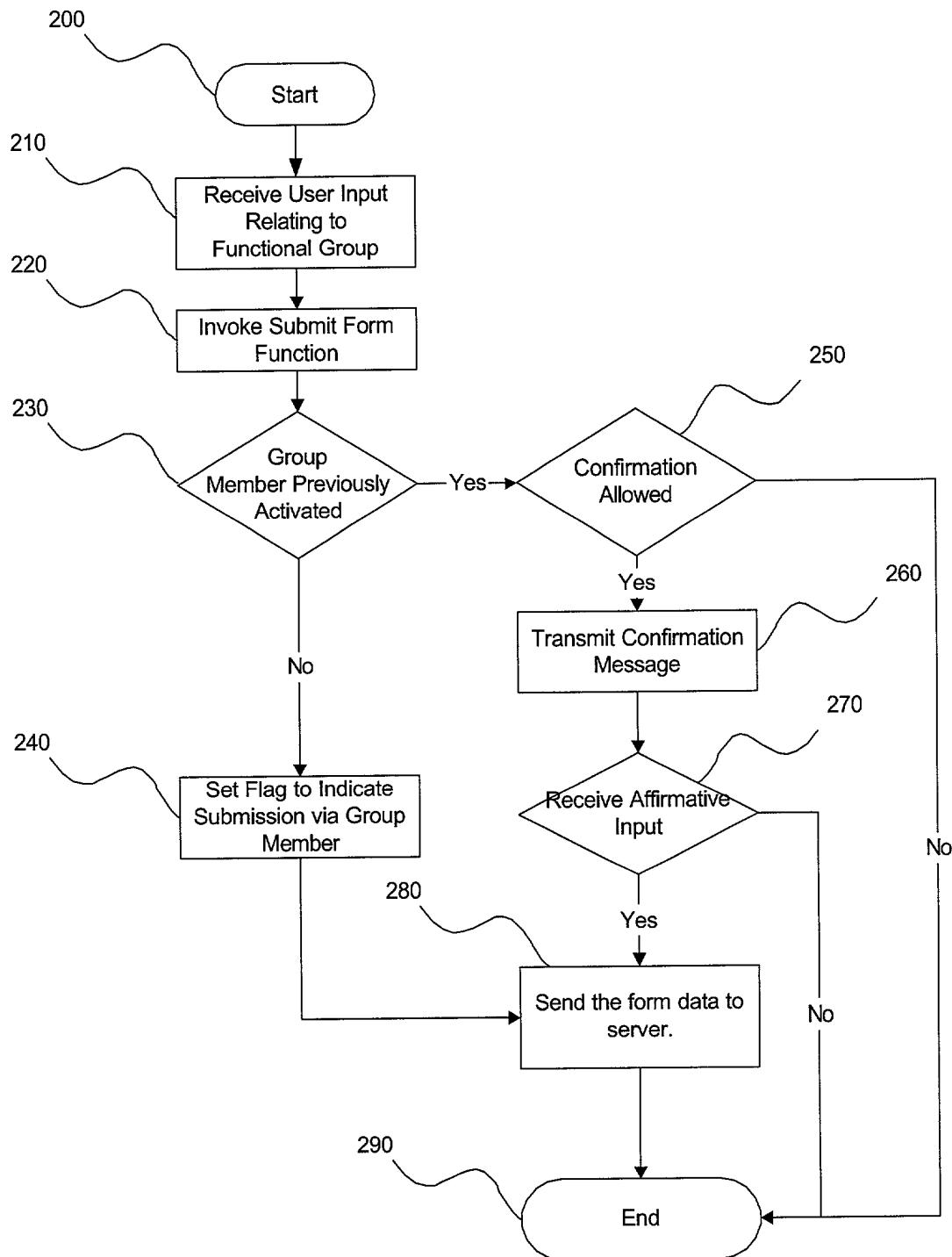


FIG. 2

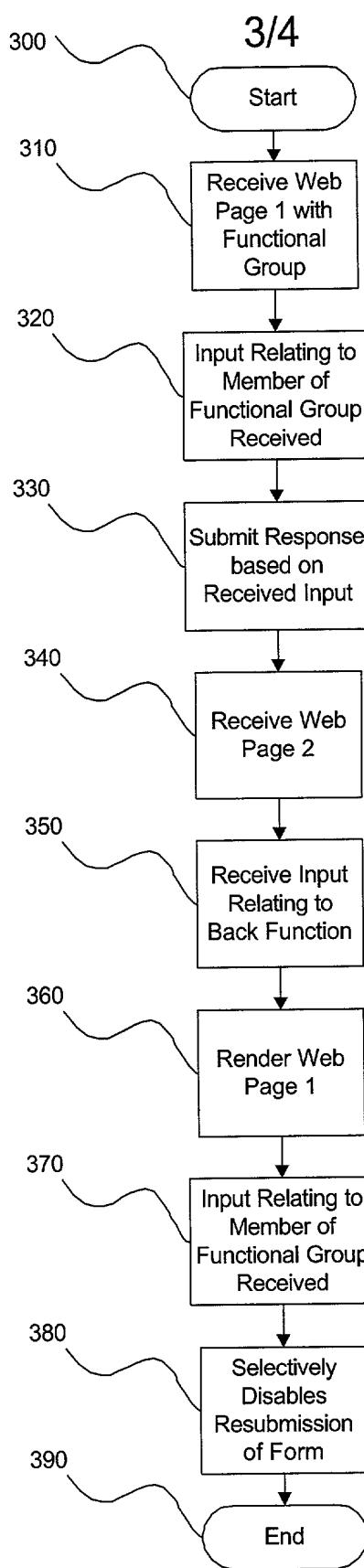


FIG. 3

```

=====
PageHistory methods

5      Description:
          definition of the PageHistory methods

=====
function PageHistory_setPageFlag(name, flag) {
10     //variable used in the cookie: literal string with ' to escape the potential : inside of name
     var namelit = "" + name + "";
     //set it in the cookie string
     //case in it is in pages already, then modify value in cookie
     if (!this.isNotInHistory(name)) {
15         var repl = new RegExp(namelit + ":" + "[^\s]*\s");
         debug(GenDebug, "this.cookie before", this.cookie);
         debug(GenDebug, "repl", repl);
         this.cookie = this.cookie.replace(repl, namelit + ":" + flag + " ");
         debug(GenDebug, "this.cookie after", this.cookie);
     }
20     else {
         //case cookie empty, create it
         if (this.cookie == "") {
             this.cookie = "{ " + namelit + ":" + flag + " }";
         }
         //else add it to the beginning
         else {
             this.cookie = "{ " + namelit + ":" + flag + " , " + this.cookie.substr(1);
         }
         this.size++;
         debug(GenDebug, "this.size", this.size);
         //case size > maxsize, remove lru = end
         if (this.size > this.maxsize) {
             debug(GenDebug, "this.cookie before size restraint", this.cookie);
             this.cookie = this.cookie.slice(0, this.cookie.lastIndexOf(",") + ")");
             debug(GenDebug, "this.cookie after size restraint", this.cookie);
         }
     }
25     debug(GenDebug, "this.cookie end", this.cookie);
     //set it in the hashtable
     this.pages[name] = flag;
     //set the cookie
     setCookie("pageh", this.cookie);
40 }
45     function PageHistory_getPageFlag(name) {
         if (!IsDef(this.pages[name])) return this.pages[name];
     }
     function PageHistory_isNotInHistory(name) {
         if (!IsUndef(this.pages[name])) return true;
50         return false;
     }
=====
55     PageHistory()
      Description:

```

FIG. 4

definition of the PageHistory constructor
The PageHistory object stores a mru/lru (most and least recently used) list of size
maxSize pages for which you submitted a form

5 When you add a page and the list is full we discard lru and add it as mru
Pages are characterized by a guid and a flag
The flag indicates if the page has already been submitted

This object has some methods that lets you add a page, update a flag for a page
and get a flag from a page.

```
10 ===== */
//no need to use prototypes since we use only one instance of the object
function PageHistory(maxsize) {
    //define the member variables
    this.maxsize = maxsize;
15    this.size = 0;
    //we store a js expression defining an associative array in the cookie
    //we instantiate the array for quick lookups in the js object pages
    //we keep the cookie to manage the lru/mru for serialization
    //this makes the whole thing much faster: remove beginning and add to the end are very easy ops
20    on strings
        //and we don't have to loop to serialize/deserialize
        this.cookie = getCookie("pageh");
        debug(GenDebug, "this.cookie", this.cookie);
        if (this.cookie == "") {
            this.pages = new Object();
        }
        else {
            this.pages = eval("bozo = " + this.cookie);
            this.size = this.cookie.split(',').length;
            debug(GenDebug, "this.size", this.size);
        }
        //define the methods
        this.setPageFlag = PageHistory_setPageFlag;
        this.getPageFlag = PageHistory_getPageFlag;
        this.isNotInHistory = PageHistory_isNotInHistory;
        debug(GenDebug, "pages", dumpObject(this.pages));
    }
}
30
35
40
var pageHistory = new PageHistory(20);
```

FIG. 4

```

/*
allowSubmit(name)

 5   Description:
      checks if a page is allowed to be submitted. If it is allowed to do so, returns true and sets its
flag to 0 so that it won't be allowed again.
      to be called on a onSubmit handler
      ===== */
10  function allowSubmit(name, confirmFlag) {
    uKey = pageHistory.getPageFlag(name)
    debug(GenDebug, "uKey", uKey);
    if (uKey == "1") {
        pageHistory.setPageFlag(name, "0");
15    return true;
    } else {
        if (confirmFlag) {
            return confirm("This form has previously been submitted. Submitting again may result in an
error. Do you want to submit?");
        } else {
            alert("This Form has previously been submitted. It cannot be submitted again.");
            return false;
        }
    }
25 }

/*
putPageInHistory(name)

30   Description:
      puts a page in history if it is not there, with a flag allowing it to be submitted.
      To be called at the beginning of your page
      ===== */
35  function putPageInHistory(name) {
    debug(GenDebug, "pageHistory.isNotInHistory(name)", pageHistory.isNotInHistory(name));
    if (pageHistory.isNotInHistory(name)) {
        pageHistory.setPageFlag(name, "1");
    }
40 }

function dumpObject(obj) {
    var dump = "";
    for (var i in obj) dump += i + "=" + obj[i] + "\n";
45 }

```

FIG. 4

```

/*
  getObject()

  Description: convert object name string or object reference into a valid object reference
  for both browsers: this is a reference on which you can set some style attributes
  */

function getObject(obj) {
  var theObj;
  if (typeof obj == "string") {
    var iniObj;
    if (isNav6) {
      iniObj = document.getElementById(obj);
    }
    else {
      iniObj = eval("document." + coll + obj);
    }

    if (!IsUndef(iniObj)) {
      return "undefined";
    }

    if (isNav4) {
      return iniObj;
    }
    else {
      // in the IE or NS6 case the iniObj.style object may be undefined
      if (!IsDef(iniObj.style)) {
        return iniObj.style;
      }
      else {
        return "undefined";
      }
    }
  }
  else {
    theObj = obj;
  }
  return theObj;
}

```

FIG. 4

```
=====
5      getObjectRef()
6
7      Description: convert object name string or object reference into a valid object reference
8      for both browsers, without the style in IE: this is the real object reference
9      this function is adapted from Danny Goodman's "Dynamic Html : The Definitive Reference"
10     http://www.amazon.com/exec/obidos/ASIN/1565924940/qid%3D963012863/002-0174003-
11     8509633
12
13 ===== */
14
15     function getObjectRef(obj) {//alert("getRef "+obj);
16
17         var theObj;
18         if (typeof obj == "string") {
19             var iniObj = eval("document." + coll + obj);
20             //alert("getRef "+iniObj);
21             if (IsUndef(iniObj)) {
22                 return "undefined1";
23             }
24             return iniObj;
25
26         }
27         else {
28             theObj = obj;
29         }
30         return theObj;
31     }
32
33 }
```

FIG. 4

```

/*
=====
FUNCTION:  IsUndef

5    INPUT:      val - the value to be tested
    RETURN:     true, if the value is undefined
                 false, otherwise.

10   PLATFORMS: Netscape Navigator 3.01 and higher,
                 Microsoft Internet Explorer 3.02 and higher,
                 Netscape Enterprise Server 3.0,
                 Microsoft IIS/ASP 3.0.
===== */

15   function IsUndef( val ) {
    var isValid = false;
    if (val+"" == "undefined")
        isValid = true;

20   return isValid;
} // end IsUdef

25   function IsDef( val ) {
    return !IsUndef(val);
} // end IsUdef

30   /**
* <pre>
* This function
*     checks if the form is allowed to be submitted.
*     if yes, it sets the action, then calls form.submit()
*
* Usage:
*
* <code>
*     <a href="Javascript:submitForm('/iMM/somefile.jsp', 'myForm', 'someUniqueKey', true)">
* </code>
*
* </pre>
*/
40   function submitForm(action, formName, key, confirmFlag) {
    //we put a provision here to let the users of this function not provide the last argument, which the
    //defaults to true
    if (IsUndef(confirmFlag)) {
        confirmFlag = true;
    }
    if (allowSubmit(key, confirmFlag)) {
        debug(GenDebug, "formName", formName);
        eval("document." + formName + ".action=\\" + action + "\\\"");
        eval("document." + formName + ".submit()");
    }
} // end submitForm

```

FIG. 4